ISO TC184/SC4/WG10 N317
Date: 2000-11-01

ISO TC 184/SC4 DRAFT STANDING DOCUMENT

Technical Committee 184 for Industrial Automation Systems and Integration Subcommittee 4 for Industrial Data

Guidelines for the content of application modules Revision 0.7

ISO TC 184/SC4 WG10 Technical Architecture
National Institute of Standards and Technology
Building 220/Room A127
Gaithersburg, Maryland 20899
USA

Contents

Foreword

The International Organization for Standardization (ISO) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

This standing document was prepared by Technical Committee ISO/TC 184, *Industrial automation systems and integration*, Subcommittee SC4, *Industrial data*.

ISO/TC 184/SC4 standards are prepared according to guidelines put forth in the following standing documents:

- Guidelines for the content of application modules;
- Guidelines for application interpreted model development;
- Guidelines for the content of application protocols using application modules;
- Guidelines for the development of abstract test suites;
- ISO/TC 184/SC4 organization handbook;
- Supplementary directives for the drafting and presentation of ISO 10303.

The use of the following guidelines are deprecated due to the new guidelines for application modules and application protocols using application modules:

- Guidelines for the development and approval of STEP application protocols;
- Guidelines for the development of mapping tables;
- Guidelines for application interpreted construct development.

Introduction

ISO 10303 is an International Standard for the computer-interpretable representation and exchange of product data. The objective is to provide a neutral mechanism capable of describing product data throughout the life cycle of a product independent from any particular system. The nature of this description makes it suitable not only for neutral file exchange, but also as a basis for implementing and sharing product databases and archiving.

This International Standard is organized as a series of parts, each published separately. The parts of ISO 10303 fall into one of the following series: description methods, integrated resources, application interpreted constructs, application modules[1], application protocols, abstract test suites, implementation methods, and conformance testing. The series are described in ISO 10303-1.

The purpose of this standing document is to provide guidelines for the content of ISO 10303 application modules (AMs) that are used as the data specification for ISO 10303 application protocols (APs).

Application modules are the key component of the modularization of the initial ISO 10303 architecture. The modularization approach extends the application interpreted construct (AIC) concept of the initial ISO 10303 architecture through inclusion of the relevant portions of the AP's application reference model. The basis of the approach is understanding and harmonizing the requirements, both new and those documented in existing APs, grouping the requirements into reuseable modules, documenting the modules, and using the modules in the development of an application protocol. With much of the content of the initial ISO 10303 architecture AP now documented in AMs, the role of the AP is to select and constrain a set of the more generic AMs to satisfy information requirements in a particular application context.

The development of an application protocol modularization strategy was driven by several requirements from different sources:

- to reduce the high cost of developing an application protocol;
- to ensure the ability to implement a combination of subsets of multiple APs or to extend existing APs to meet a business need;
- to ensure the ability to reuse application software developed to support one AP in the development of an implementation of another AP with the same, or similar, requirements;
- to avoid the duplication and repeated documentation of the same requirements in different application protocols leading to potentially different solutions for the same requirements; and
- to ensure the ability to reuse data generated by an implementation of one or more APs by an implementation of one or more different APs.

The expected audience for this document includes developers of ISO 10303 application modules and application protocols as well as users of application protocols who are interested in a more in-depth understanding of the origins of the structure of application protocols.

NOTE - This document is an adaptation of the guidance found in *Guidelines for the development and approval of STEP application protocols*, *Guidelines for application interpreted model development*, *Guidelines for the development of mapping tables*and *Guidelines for application interpreted construct development.*

[1] To be added to ISO 10303-1 as part of the WG10 STEP Modularization PWI responsibilities.

Guidelines for the content of application modules

1 Scope

This standing document specifies guidelines for the content of ISO 10303 application modules.

The following are within scope of this standing document:

- description of the content of an application module.

The following are outside the scope of this standing document:

- guidelines for the development process for application modules;
- guidelines for the development process for application protocols using application modules;
- specification of presentation information for the documentation of an ISO 10303 application module;
- detailed guidance on how to select constructs of the ISO 10303 integrated resources that map to the information requirements of an ISO 10303 application module;
- guidelines for development of mapping specifications for documents other than ISO 10303 application modules;
- guidelines for the use of EXPRESS in information models other than ISO 10303 application modules;
- description of how the application modules are to be used in the documentation of application protocols.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this standing document. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standing document are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. Members of the IEC and ISO maintain registers of currently valid International Standards.

ISO 10303-1, *Industrial automation systems and integration — Product data representation and exchange — Part 1: Overview and fundamental principles.*

ISO 10303-11:1994, *Industrial automation systems and integration — Product data representation and exchange — Part 11: Description methods: Language reference manual* and its Amendment 1.

The following documents contain provisions which, through reference in this text, constitute provisions of this standing document. At the time of adoption, the revisions of the documents indicated were valid. All documents are subject to revision, and users of this standing document are encouraged to investigate the possibility of applying the most recent editions of the documents indicated below.

ISO/TC 184/SC4 N534:1997, *Guidelines for application interpreted construct development.*

ISO/TC 184/SC4 N535:1997, *Guidelines for the development and approval of STEP application protocols.*

ISO/TC 184/SC4 N318:2000, *Guidelines for the content of application protocols using application modules.*

ISO/TC 184/SC4 N1029, *Guidelines for the development of mapping specifications, 2nd Edition.*

ISO/TC 184/SC4 N858, *Supplementary directives for the drafting and presentation of ISO 10303, edition 2.*

3 Terms, definitions and abbreviations

3.1 Terms defined in ISO 10303-1

For the purpose of this standing document, the following terms defined in ISO 10303-1 apply.

- application reference model (ARM);
- interpretation;
- resource construct;
- unit of functionality (UoF).

3.2 Definitions

For the purposes of this standing document the following definitions apply.

3.2.1
application module (AM)

a reuseable collection of scope statement, information requirements, mappings and module interpreted model that supports a specific usage of product data across multiple application contexts.

3.2.2
module interpreted model (MIM)

an information model that uses the common resources necessary to satisfy the information requirements and constraints of an application reference model, within an application module.

3.3 Abbreviations

For the purposes of this standing document, the following abbreviations apply.

AIC application interpreted construct

AM application module

ARM application reference model

MIM module interpreted model

UoF unit of functionality

URL universal resource locator

4 Application module content overview

This clause provides an overview of the contents of an application module. The contents for an application module are given in figure 1 and are explained in the subsequent subclauses. The three major components of an AM are: 1) the scope and functional requirements; 2) the application reference model as a representation of the application domain information requirements; and 3) the module interpreted model that specifies the required use of the common resources. Additionally, each application has an associated module validation results document, test cases or abstract test suite.

Figure 1 - Contents of an application module

A AM MIM short names
B Information object registration
C ARM EXPRESS-G
D MIM EXPRESS-G
E AM ARM and MIM EXPRESS listings
F Concepts required from other AMs
G Application module implementation and usage guide
H Technical discussions
I Bibliography
Index

4.1 The foreword and introduction

The Foreword for the AM shall contain the text specified in Figure 2 replacing the items in brackets as appropriate for the AM.

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75% of the member bodies casting a vote.

In other circumstances, particularly when there is an urgent market requirement for such documents, a technical committee may decide to publish other types of normative documents:

- an ISO Publicly Available Specification (ISO/PAS) represents an agreement between technical experts in an ISO working group and is accepted for publication if it is approved by more than 50% of the members of the parent committee casting a vote;

- an ISO Technical Specification (ISO/TS) represents an agreement between the members of a technical committee and is accepted for publication if it is approved for publication if it is approved by 2/3 of the members of the committee casting a vote.

An ISO/PAS or ISO/TS is reviewed every three years with a view to deciding whether it can be transformed into an International Standard

ISO/TS 10303-28 was prepared by Technical Committee 184, *Industrial automation systems and integration*, Subcommittee SC4, *Industrial data*.

This International Standard is organized as a series of parts, each published separately. The parts of ISO 10303 fall into one of the following series: description methods, integrated resources, application interpreted constructs, application modules, application protocols, abstract test suites, implementation methods, and conformance testing. The series are described in ISO 10303-1[1]. A complete list of parts of ISO 10303 is available from the Internet:

<http://www.nist.gov/sc4/editing/step/titles/>.

This part of ISO 10303 is a member of the application modules series.

Annexes <normative annex list> form an integral part of this part of ISO 10303. Annexes <informative annex list> are for information only.

1 - A future edition of ISO 10303-1 will describe the application modules series.

Figure 2 - The boilerplate text for the Foreword of an application module.

The Introduction for the AM shall provide an overview of the technical content. The Introduction shall explain the relationships between the AMs that are used by the AM being defined and shall begin with the text specified in Figure 3.

ISO 10303 is an International Standard for the computer-interpretable representation of product information and for the exchange of product data. The objective is to provide a neutral mechanism capable of describing products throughout their life cycle. This mechanism is suitable not only for neutral file exchange, but also as a basis for implementing and sharing product databases, and as a basis for archiving.

This application module <application module-specific introductory text>.

Figure 3 - The boilerplate text for the Introduction of an application module.

4.2 The scope

Clause 1 of an AM shall define the domain of the AM and summarize the fundamental concepts and assumptions of the scope, the functionality of the AM, and the types of information that are accommodated by the AM. A description of the functionality and information that are specifically outside the scope of the application module shall be defined to clarify the domain of the AM.

4.3 The normative references

All normative references shall be listed in clause 2 of an AM. The minimal required set of

normative references are:

ISO 10303-1 Industrial automation systems and integration - Product data representation and exchange - Part 1: Overview and fundamental principles.

ISO 10303-11 Industrial automation systems and integration - Product data representation and exchange - Part 11: Description methods: The EXPRESS language reference manual.

ISO/IEC 8824-1:1995, Information Technology - Open Systems Interconnection - Abstract Syntax Notation One (ASN.1) - Part 1: Specification of Basic Notation.

The normative references shall include the application modules directly used by the application module being defined.

4.4 The definitions and abbreviations

Clause 3 of an AM shall include definitions of all concepts necessary to understand the Introduction, Scope, and Information requirements clauses. This clause may include concepts that are defined further in the Information requirements clause. The concept definitions provided in this clause shall be consistent with the complete definitions provided in the Information requirements clause. This clause shall not include the definitions of objects defined in the application reference model or the module interpreted model. This clause shall list the terms defined in other ISO standards, including AMs, that are necessary for understanding the AM.

4.5 The information requirements

Clause 4 of an AM shall describe the functionality and information requirements of the AM. The first paragraphs of this clause provide a high level description of the information requirements that are supported by the AM and a summary of the structure used to partition the information requirements.

This clause may include a description of the types of information supported by the AM, any restrictions on the information supported, and the supported uses of the defined information.

This clause shall provide all additional information on the fundamental concepts and assumptions (initially introduced in clauses 1 and 3) which is necessary for complete understanding of the information requirements and the scope boundaries. This clause shall include a description the fundamental concepts and UoFs of the AM and any prerequisite AMs used by the AM.

This clause shall include subclauses for units of functionality, referenced AM ARMs and ARM type definitions, ARM entity definitions, ARM rule definitions, and ARM function definitions as required. The ARM shall be defined using the EXPRESS language and is constructed from AM specific declarations and other AM ARMs referenced using the EXPRESS interface specification (USE FROM) defined in ISO 10303-11.

4.5.1 The units of functionality

If the AM contains UoFs, the first subclause of clause 4 of an AM shall specify a list of the UoFs defined in or used by the AM and the definition of each UoF defined in the AM.

A UoF is a grouping of data constructs which is important in the application module. A UoF specifies the set of application objects that constitute one or more concepts of the application reference model. UoFs are a mechanism for modularizing the information requirements into primary concepts. The UoFs are used to organize and summarize the functionality of the ARM.

Each UoF definition shall include the scope of the UoF, a description of the function(s) that the grouping of data is intended to support, and a list of the application objects that are included in the UoF.

For a UoF defined in another application module the UoF description in the using AM shall list the application objects referenced by objects in the using AM.

NOTE - The intent is for application modules to define one UoF and possibly make use of UoFs defined within other application modules.

4.5.2 The required application module ARMs

The next subclause of clause 4 of an AM shall specify the application reference models defined in other application modules that are required by the AM. This specification takes the form of documented EXPRESS USE FROM constructs and shall use the ARM from the required AMs in its entirety.

As part of the migration path to a modularized ISO 10303 standard, consensus may be reached on the scope of an AM under development which requires other AMs that do not yet exist. In this case the AM under development may create an informative ARM defining the concepts it requires and document that ARM in annex F. Requirements on an interpretation of that ARM may also be specified in annex F.

4.5.3 The ARM type definitions

If the AM ARM declares defined types, the next subclause of clause 4 of an AM shall specify the defined types necessary for the description of the application objects defined in the AM.

4.5.4 The application objects as ARM entity definitions

The next subclause of clause 4 of an AM shall include the definitions for all application objects and attributes declared in the AM. An application object is an atomic element of an application reference model that defines a unique application concept and contains attributes specifying the data elements of the object. Application objects are documented in the ARM by EXPRESS ENTITY definitions.

Application objects documented as EXPRESS entities contain the definitions of simple attributes, derived attributes, relationship attributes and relationship constraints. Simple attributes are those which evaluate to a simple data type. Relationship attributes are those which establish a relationship with another application object. Relationship constraints are EXPRESS INVERSE attributes which constrain the cardinality of a relationship between two application objects.

4.5.5 The ARM rule definitions

If the AM ARM declares rules, the next subclause of clause 4 of an AM shall specify the

necessary rules constraining the application objects.

NOTE: Global rules should be examined carefully to avoid unnecessarily preventing the reuse of the AM.

4.5.6 The ARM function definitions

If the AM ARM declares functions, the next subclause of clause 4 of an AM shall specify the necessary functions used by the ARM rules and ARM entity definitions.

4.6 The module interpreted model

Clause 5 of an AM shall specify the module interpreted model. The MIM shall be defined using the EXPRESS language and is constructed from the resource constructs (including the MIM schemas of other AMs) using the EXPRESS interface specification (USE FROM) defined in ISO 10303-11. The required resource constructs may be further refined in the AM. The need for refinement of the resource constructs arises out of the information requirements of the particular application module domain.

4.6.1 The mapping specification

Clause 5.1 of an AM shall specify the mapping specification. The mapping specification documents the correspondence between the information requirements and the constructs of the MIM. This mapping specification shall specify an unambiguous mapping between the application objects and their attributes defined in the information requirements clause and the constructs of the MIM. The mapping shows how the common resource constructs are used to meet the information requirements of the application module.

4.6.1.1 The reuse and specialization of mapping specifications

When one AM requires the use of another AM that use includes the mapping specification of the used AM in the scope of the using AM. This reuse of mapping specifications is the same as the reuse of the ARM and MIM from a used AM in the scope of the using AM. The using AM may not contradict the mapping specification in the used AM but may specialize the mapping specification to meet more specialized requirements defined in the using AM. This specialization of a mapping specification consists of the following:

- the elimination of one or more possibilities in an OR map;
- the addition of mapping constraints on attribute domain values.

4.6.1.2 The mapping specification for abstract concepts in the ARM

In the case that a generic and abstract concept appears in the ARM of an AM with the expectation that the concept will be completed in the ARM of a using AM, then a mapping specification need not be complete for that ARM concept.

Additionally, if a mapping specification is written for a concept, the concept need not appear in the MIM short form schema. This allows for the concept to be completed in a using ARM where a complete mapping specification and instantiable and complete MIM short form constructs are defined. An example of the use of this mechanism might be to address Part 41 management

resource assignment completion.

4.6.2 The MIM short form

Clause 5.2 of an AM shall specify the MIM EXPRESS short listing. The MIM EXPRESS short listing shall consist of USE FROM statements that select common resource constructs and the MIM schemas from other AMs, AM specific declarations, and any appropriate modifications to textual material that applies to constructs interfaced into the MIM schema from the common resources. The declarations include TYPE declarations, ENTITY declarations that create subtypes of resource entities, and any necessary RULES, and FUNCTIONS that are required to satisfy the information requirements. Any declarations of types, entities, rules, and functions defined in the AM are fully documented in the MIM EXPRESS short listing. Textual modifications may also be made in the MIM documentation including the following:

- clarification of application specific interpretation of the meaning of a generic entity definition;
- clarification of application specific interpretation of the meaning of one or more attributes;
- specification of application specific informal propositions;
- addition of application specific examples and notes.

4.7 The annexes and bibliography

The following annexes appear in an AM.

A MIM short names (normative)

> This annex shall contain a correspondence list between the entities used in the MIM and the short names. This list is derived from the short names specified in the common resources together with the short names for entities introduced in the AM.

B Information object registration (normative)

> This annex shall specify the information object identifiers for the application module. This shall include identifiers for the AM document, the ARM schema and for the MIM schema.

C ARM EXPRESS-G (informative)

> This annex shall contain an EXPRESS-G representation containing at a minimum all types and entities defined in the ARM. Additional constructs from the ARMs defined in required AMs may also be contained in order to assist in clarifying the relationships between ARM concepts. The graphical presentation of the ARM, i.e., EXPRESS-G, aids the understanding and review of the information requirements and definitions. The ARM diagrams shall be at a detail level sufficient to present the requirements in a manner that it is understandable to an application domain expert. This representation shall be documented in accordance with annex D of ISO 10303-11 and the *Supplementary directives for the drafting and presentation of ISO 10303*.

D MIM EXPRESS-G (informative)

This annex shall contain the EXPRESS-G representation of all interface specifications and all types and entities defined in the MIM. This representation shall be documented in accordance with annex D of ISO 10303-11 and the *Supplementary directives for the drafting and presentation of ISO 10303*.

E MIM and ARM EXPRESS listing (informative)

This annex shall contain a reference to a URL with the entire MIM and ARM EXPRESS short listing without comments.

F Concepts required from other AMs (optional and informative)

This annex, if provided, contains the concepts required from other AMs that do not yet exist. These include one or more annotated informative ARM schema declaring constructs required by the ARM schema of the AM being defined. Constraints on the interpretation of the informative ARM schema may also be defined. These define the concepts required for the successful use of the AM without defining them completely and without specifying where they will be standardized.

G Application module implementation and usage guide (optional and informative)

This annex, if provided, contains informative guidance on implementing and using the AM. This annex provides guidance to two different audiences, i.e., implementors and end users of AM compliant implementations. Example information descriptions that are supported by the AM and the corresponding AM exchange files may be included in this annex. If exchange files are included in this annex, the annex should explain the primary data structures and the logic and meaning of the values used in the exchange file.

H Technical discussions (optional and informative)

This annex, if provided, contains a summary of relevant technical discussions and the resolution of issues raised during the development of the AM. This annex provides background information for potential users of the AM and for developers of similar or related AMs. The material given should not cast doubt or self justify. Only material which supports the normative text shall be given.

Bibliography (informative)

This lists all informative references relevant to the AM.

5 Specification of application module content

This clause specifies the required content for an AM. Unless overridden by this standing document, the presentation of the various elements of an application module is governed by the relevant ISO directives and SC4 standing documents.

5.1 Scope content

The definition of the scope and information requirements begins with the formulation of a statement of the application module functional requirements. The detailed scoping and information requirements definition shall follow from this statement.

The scope and requirements identify the primary concepts and relationships to be supported by the AM. The AM's scope and information requirements shall be carefully defined and documented. The AM scope statement shall include a summary of the data that are within scope. This scope statement shall define the following characteristics of the scope of the application module as appropriate given the AM domain:

- types of information;
- types of data;
- life cycle stages supported;
- uses of the information, e.g., functional or application processes, supported;
- discipline views supported.

For the purposes of clarification, an AM exclusion from scope statement shall appear that includes the data outside the scope of the AM. The same characteristics used to define the scope may be included as appropriate.

5.1.1 Scope refinement

Scope refinement is the process of taking a single application module and breaking it down into two or more application modules to address exactly the original scope. A revision is then made to the original AM such that it uses the new AMs to meet its requirements. The refinement of the scope of an AM includes identifying and separating concepts to maximize reuse of the AMs and identifying mandatory relationships with other AM's. Each AM should define a unique and distinct set of Application Objects relating only to the concept being represented. It should contain no Application Objects that are defined by another AM, that is representing another concept. Vocabulary will typically change from concept to concept. The description of the AM should not contain sentences that move from one concept to another. The use of conjunctions, such as 'and', in either the name or description of an AM is an indication that there is more than one concept being represented. Unless it can be demonstrated that the concepts being represented are inseparable in all applications, serious consideration should be given to separating those concepts and defining an AM for each.

5.2 Information requirements specification

Based on the detailed scope and functional requirements that have been defined, the information domain of the AM is defined in the application reference model (ARM). The ARM shall be documented using EXPRESS. The ARM shall describe fully the data needs of the application module, using the potentially harmonized terminology of the application domain(s).

The ARM documents the required data and relationships of the AM. The information

requirements shall be modelled only to the level necessary to convey the information that is important from the application experts' point of view. An ARM shall be sufficiently detailed so that the selection and interpretation of the common resources can be done accurately.

A mechanism for modularizing the scope of an industry domain into manageable constructs is to define units of functionality. A UoF is a collection of application objects and assertions that conveys one or more well-defined concepts within the context of an ARM. A UoF may result in one or more AM's and an AM may include or use more than one UoF. However, to realize the most reusability of an application module the scope of application modules that define multiple UoFs should be reviewed for potential refinement. A UoF usually supports an application function or process. UoFs are used to organize and summarize the functionality of the ARM. For example, if a geometric modelling application has a requirement for wireframe geometry, then a UoF may be defined which provides a grouping of those application objects in the ARM which are intended to support geometric modelling using wireframe geometry.

In a harmonized suite of application modules, two or more AMs shall not contain equivalent UoFs or common information requirements. When two or more AMs have equivalent UoFs or common information requirements, a new AM shall be created, the same interpretation of the integrated resources shall be used in the new AM, and the new AM shall be used by the AM's with common requirements without changing the scope of the existing AM.

5.2.1 Information requirements documentation

Clause 4 of the AM shall include a high level description of the information requirements, a summary of the structure used to define the partition of the information requirements defined by the AM, and subclauses for specifying UoFs, referenced AMs and the AM ARM components. The description of the information requirements shall be sufficient to prepare the reader for the material in the subclauses.

The information requirements shall be defined using EXPRESS annotated with prose. The referenced AMs and the ARM components including application objects, relationship attributes and constraints shall be defined as a single documented EXPRESS schema. The elements listed within each subclause shall be ordered alphabetically. In a harmonized suite of application modules, the UoFs, ARM schema, ARM types, application objects, ARM rules and ARM functions shall have unique names, i.e., no application elements shall share the same name across the complete suite of AMs.

The documentation for an AM's ARM and information requirements includes the following components as required. The clause referenced listed assume at least one of each component is included. Should there be no ARM types, functions or rules these clauses shall be omitted in the AM and the clause numbering shall change accordingly.

1. units of functionality

This subclause provides a complete list of the UoFs defined in the AM and defined in any other AM used directly or indirectly by the AM. This is the only place in an application module where the full domain of the application module and all of its prerequisite application modules is found. For UoFs defined in the AM, a description of the functions that each UoF supports, and the list of application objects included in the UoF is specified. For UoFs used from another AM, the

name of the AM in which which the UoF is defined shall be included in the list. Application objects defined in another AM specifically referenced by application objects defined in the AM shall be listed under the description of the UoF in the other AM in which they are defined.

2. referenced application module ARMs

The referenced application module ARMs takes the form of documented EXPRESS USE FROM constructs. Each AM ARM shall be used in its entirety. Should the AM not use any other AM this fact shall be stated in this subclause.

3. ARM type definitions

The ARM defined types specification takes the form of documented EXPRESS TYPE definitions.

4. ARM entity definitions

The application objects are documented as ARM EXPRESS entities. Every EXPRESS entity in the ARM shall be considered an application object. Each application object which exists in the ARM shall appear in the mapping specification although it may not be fully mapped.

5. ARM rule definitions

The ARM rules specify the necessary constraints on the application objects. These constraints are documented in the ARM as EXPRESS RULE definitions.

6. ARM function definitions

The ARM functions specify the necessary functions used by the ARM rules and ARM entity definitions. These functions are documented as EXPRESS FUNCTION definitions.

5.2.2 EXPRESS ARM documentation

The ARM EXPRESS definitions shall be specified as follows.

- The ARM schema name shall be the name of the AM appended with the suffix "_arm".
- The ARM type definitions shall appear in alphabetical order. Each type shall appear in its own subclause.
- Each application object shall be stated in the ARM entity definitions and each ARM entity definition shall represent an application object. Each application object shall appear in its own subclause.
- Each attribute whose data type is either a base data type or a defined data type which is a SELECT data type with a select list that does not contain entity types or, recursively, other SELECT types with select lists that do not contain entity types shall be stated as an attribute of that entity in the ARM entity definition.
- Each attribute whose data type is an aggregate with a type that is either a base type or a defined type which is a SELECT data type with a select list that does not contain entity

types or, recursively, other SELECT types with select lists that contain entity types shall be defined as an attribute in the ARM entity definition, with the cardinality specified in the definition.

- Each relationship attribute whose data type is an aggregate of either an entity type or a SELECT type with a select list that contains either entity types or other select types with select lists that contain entity types shall defined as an attribute in the ARM entity definition with the cardinality defined by the aggregate bounds.

- Each relationship attribute whose data type is an entity type shall be defined as an attribute in the ARM entity definition.

- Each relationship attribute whose data type is a SELECT data type with a select list that contains entity types or other select types with select lists that contain entity types shall be defined as an attribute of the ARM entity definition.

- Each constraint on the cardinality of a relationship attribute whose data type is an entity type shall be defined as an INVERSE attribute in the referenced ARM entity definition.

- The ARM rule definitions shall appear in alphabetical order. Each rule shall appear in its own subclause.

- The ARM function definitions shall appear in alphabetical order. Each function shall appear in its own subclause.

5.3 The module interpreted model

The MIM documents the interpretation of the the information requirements into the common resources. The results of that interpretation are:

- a selection of the required application module MIMs to satisfy ARM requirements;

- the selection of the required SC4 common standardized resource constructs to satisfy ARM requirements;

- additional EXPRESS constructs and constraints needed to satisfy ARM requirements and to specify the MIM short listing, and;

- the mapping of ARM constructs to the MIM schema constructs.

5.3.1 Selected resource constructs

The result of interpretation is the specification of one or several resource constructs which satisfy the requirements of the ARM construct, along with any needed constraints.

The integrated resource constructs are designed for generic use by all AMs. In the selection process, the best entity for an ARM requirement may have attributes that are not supported by requirements in the ARM. In the cases where the additional attributes have underlying types that are base types and there is no data to support the instantiation of these attributes, the recommended values for these attributes may be documented in the "Module implementation and

usage guide" annex of the application module.

5.3.2 MIM short form specification

The MIM is constructed from the common resources, including AMs, through the use of the EXPRESS USE FROM; this MIM schema is called the short form. The short form consists of two parts. The first part contains the interface specification that provides the relationship between the common resources and the MIM. The second part defines the unique MIM constructs that refine or specialize the usage of the integrated resources. The MIM schema name shall be the name of the AM appended with the suffix "_mim". The remainder of this subclause of this standing document expands on the detailed use of EXPRESS in defining an MIM.

The EXPRESS short listing specifies the selection of application module MIMs and constructs from other common resources through a formal interface specification method. This interface is accomplished through the EXPRESS USE FROM specification described in clause 11 of ISO 10303-11. The required MIMs from other AMs shall be used in their entirety.

The MIM short listing also specifies all constructs that are new and unique to the MIM, which may include entities, attributes, type definitions, local and global rules, and functions. Only two classes of new constructs are allowed in an MIM; those that:

- complete and assign a concept definition; and
- constrain a generic concept.

5.3.3 Concept completion and assignment

In the ISO 10303 integrated resources, there are a number of template structures that have been used to specify product data management resources, such as the construct specified in annex E of ISO 10303-41. These template structures are semantically incomplete by themselves; they are designed to be used to specify structurally similar though semantically dissimilar concepts in a standard and consistent manner. The template entity is completed by an explicit and unique assignment of semantics in the MIM. The explicit assignments are made between the management concepts (e.g., approval or organization) and the items that require the management information. Details on how the new MIM constructs which complete the assignment template structure are documented in the MIM short listing follow.

Subtypes of interfaced entities may be created for the completion and assignment of generic concepts. The assignment of a generic concept is the only time that an explicit attribute may be added to a subtype declared in an MIM schema. In order to assign the generic constructs to the appropriate entities as defined in the ARM requirements, two constructs are created. The first construct is a SELECT type. That SELECT type may contain all of those entities or other SELECT types that may have the concept assigned to it. That SELECT type may also be an extensible SELECT type. In this case, using AMs extend the SELECT type domain to complete the entities or SELECT types that may have the concept assigned to it. The second construct is an entity which is a subtype of the entity in the ISO 10303 integrated resources representing the generic concept. This entity shall contain a single attribute which references a SET of the SELECT type that was defined previously. A SET is used here so that all concepts that are the same are able to be assigned to different instances of entities that share the information. The

subtypes shall be named with the name of the entity from ISO 10303-41 prepended with the string "applied_", the attribute shall be named "items" and the SELECT type shall be named based on the entity name from ISO 10303-41. See B.4 for an example.

A mapping specification supporting the concept completion and assignment constructs defined in the AM shall be included in this clause of the AM.

5.3.4 Constrained generic concepts

Standardized common resource constructs are generic in nature and designed to be shared by multiple application contexts. Several AMs may require the same generic concept that a resource construct represents, but each may constrain the generic concepts in order to represent a specific usage within the AM domain. New MIM constructs may be created (through subtyping) to:

- constrain a generic concept;
- constrain the relationship between generic concepts; or
- create multiple, specialized concepts of the generic concept, through specification of different constraints representing different usages of the same generic concept under specific circumstances within the domain of the AM.

Global rules and functions and subtype constraints are allowed in an MIM to constrain generic concepts. In the case that a constraint in the MIM is the result of a constraint in the ARM that constrains concepts defined in a used AM, the documentation of the MIM constraint shall reference the ARM constraint upon which it is based.

Global rules are frequently written to constrain an entity from being independently instantiated.When an application requirement results in a constraint on every use of an interfaced entity or attribute of an interfaced entity, a global rule is specified. Global rules are also used to specify constraints on the relationships among two or more entities. Constraints may be specified as global rules to serve four specific functions and as subtype constraints to serve one function. These functions are described in the following subclauses.

5.3.4.1 Global rules as supertype constraints and using subtype constraints

A supertype constraint is one that constrains the relationships among interfaced entities in the same subtype/supertype tree. These constraints are in addition to any constraints applied through the specification of the interface statements (i.e., dependence). There are many uses for this type of constraint; only two uses are described here.

A supertype constraint, for example, may make the instantiation of a subtype mandatory for a particular supertype entity. The requirement for a rule of this type will arise when an entity is 1) explicitly interfaced to be subtyped, or 2) is implicitly interfaced via attribute reference as well as subtype reference. If the ARM requirement is only for the new subtype entity, then a rule is required to prohibit the instantiation of the interfaced supertype entity. See annex C for a template for a mandatory subtype global rule.

Rules for constraining supertype relationships may also be used to limit the combinations of subtypes of a single supertype entity in order to define the appropriate set of complex entity data

types allowed satisfy the requirements specified in the ARM.

In the cases where any constraint may be written using an EXPRESS subtype constraint instead of a global rule, a subtype constraint may be specified for the that purpose.

5.3.4.2 Cardinality constraint

A cardinality constraint is one that constrains the relationship between two interfaced entities by limiting the number of instances of one entity type that may be related to instances of the other. Explicit cardinality is specified in the referential direction of the relationship (i.e., from the entity with the attribute to the entity that is the data type of that attribute). When an INVERSE attribute does not exist to constrain the cardinality of the referenced entity, the cardinality defined is, by default, zero, one or many. These cardinalities may be constrained using a cardinality constraint. See annex C for a template for a global rule which constrains cardinalities of referenced entities.

The integrated resources, for example, always model the existence dependency of one entity with respect to another by the referential direction of the relationship. That is, the dependent entity always references the independent entity so that an instance of the dependent entity requires the existence of an instance of the independent entity. In the ARM, two entities might be interdependent. That is, each entity requires a single instance of the other in order to model a complete concept. In this case, a cardinality constraint would be necessary to constrain the inverse cardinality to be exactly one rather than the default zero, one or many.

5.3.4.3 Referential integrity constraint

Global rules can be used to specify referential integrity constraints as defined in 5.3.4.2.1. This type of constraint is one that constrains valid reference paths for all instances of an interfaced entity. The paths that are constrained may be combinatorial in nature. These combinatorial constraints are ones where a single entity may be required to be instantiated through two or more distinct paths in order to completely satisfy an ARM requirement. This type of constraint may also constrain an attribute value that is reached via a single reference path for all instances of a referencing entity.

5.3.4.4 Attribute domain constraint

An attribute domain constraint is one which constrains the value of an attribute in instances of a particular entity. This constraint is used to constrain the values, for example, of an attribute of type STRING in an interfaced entity to correspond to values of an enumeration as defined in ARM requirements. In the design of the integrated resources, the use of enumerated types was limited to preserve the generic nature of the integrated resources. Instead of referencing enumerated types, integrated resource entity attributes reference base types or defined types where the underlying type is a base type (i.e., INTEGER, STRING etc.). When an application context as defined in an ARM is used to determine requirements for specific structures, enumerations are considered specific requirements. A domain constraint specifies the legal values of the simple data types that correspond to the enumeration values and the standard interpretation of those values for that application context. See annex C for a template for a global rule which restricts attribute values.

Another use of an attribute domain constraint is to place limits on the values of attributes of an

entity. For example, if an entity has an attribute that is an INTEGER, and an ARM specifies a requirement that the INTEGER value must be greater than 1, the domain constraint is specified on the entity that contains that attribute to declare that constraint.

5.3.5 Use of functions

In the specification of rules, EXPRESS functions may be used in order to make the specification of the rule simpler. Due to the recursive nature of many of the references in the common resources, a constraint may need to be defined recursively. Additionally, many rules may need to use the iterative and logical capabilities of the executable statements defined in EXPRESS in order to specify complex constraints on the interactions among different entities. This type of complex interaction will usually require that a function be defined in order to support the specification of a constraint. In these cases, new functions may be defined to be used by a rule.

5.3.6 Completed short form schema

Global rules may appear in the MIM to constrain an entity, relationship or attribute.

Constructs from the integrated resources are pruned in the MIM (see 5.1.2 of *Guidelines for application interpreted model development*). There may be subtypes and items of select lists that appear in the integrated resources that are not imported into the AM. Constructs are eliminated from the subtype tree or select list through the use of the implicit interface rules of ISO 10303-11. References to eliminated constructs are outside the scope of the MIM.

5.3.7 EXPRESS usage guidelines for MIMs

This clause describes the use of the EXPRESS language in an MIM EXPRESS short listing, and provides additional details that describe the semantics of a particular usage of EXPRESS.

5.3.7.1 Schema interface

An MIM establishes a formal relationship to the integrated resources and application modules by containing the EXPRESS USE FROM keyword. USE FROM is the only interface mechanism that may be employed in an MIM.

The USE FROM keyword interfaces named data types and entire schemas into another schema. In an MIM EXPRESS short form, if the interface is to a construct defined in the integrated resources, the desired construct is individually named in the interface specification for the schema in which the construct is defined. If all constructs from an integrated resource schema are desired, the entire schema may be interfaced by providing only the schema name in the interface specification. If the interface is to an AM, only the AM schema name is provided. The AM must be used in its entirety; the use of subsets of an AM is not allowed.

5.3.7.2 Entity type specialization

The requirements for the declaration of unique MIM constructs during the development of the MIM EXPRESS short listing is defined earlier in this standing document. Entities are created in the MIM only to achieve one of three objectives: constraint localization, attribute definition specialization, and concept completion and assignment. Entities used from the Integrated Resources and from the MIM schemas in other AMs may be subtyped as part of meeting one of

these three objectives. Concept completion and assignment also requires the specification of a SELECT type. In the development of an MIM, this is the only case where an MIM-specific defined type is specified.

5.3.7.2.1 Localization of constraints

In an MIM, there may be requirements to specify constraints on an entity that differ depending on its usage in different structures. This type of constraint is called an entity behavioral constraint. In order to specify this type of constraint, the entity is interfaced explicitly into the MIM schema from the integrated resources or included via a USEd AM. There are two methods for representing the constraint:

- a global rule is defined to constrain attribute values representing the required constraint from the ARM; or
- a new subtype of the entity is be created to represent the concept for which the constraints are defined.

Using global rules is the preferred method to consider to represent these constraints. Global rules that, when pulled into a larger context, may apply to a broader scope than originally intended are discouraged. Subtyping should be employed when the complete set of subtypes for all possible domains can be determined.

In addition to constraints on the usage of a particular entity, there may be a need to specify differing, and often conflicting, constraints on an entity or an entity's attributes depending on its usage in the reference path of a generic entity interfaced to the MIM. In this case, subtypes of the referencing entity shall be created in order to establish a context for the constraint and to specify the constraint on the referenced entity. This type of constraint is called a referential integrity constraint.

5.3.7.2.2 Specialization of attribute definitions

The creation of subtypes of the interfaced entities from the integrated resources or AMs enables more specific attribute definitions to be given when the generic definition is not sufficient to satisfy the ARM requirements. This situation most often occurs when there is an ARM requirement for relationships defined between two entities that play specific roles within those relationships. There are two practices which may be used in this case, depending on the application requirements.

The first practice concerns attribute naming. If there is a requirement that an attribute have a specific name based on the ARM, then a subtype entity is created in the MIM and a derived attribute is specified which names the attribute inherited from the supertype entity. Derived attributes may be used, for example, to specialize the generic product_definition_relationship entity when an application requirement states that the relationship is prioritized: one product_definition is first priority and the other is second priority. The attributes names first_priority and second_priority have a very specific application meaning and the development team has defined those roles specifically within the ARM requirements. The MIM would then contain an entity such as:

```
*)

ENTITY priority_product_definition_relationship;
SUBTYPE OF (product_definition_relationship);
DERIVE
first_priority : product_definition :=
SELF\product_definition_relationship.relating_product_definition;
second_priority : product_definition :=
SELF\product_definition_relationship.related_product_definition;
END_ENTITY;

(*
```

The second practice is used if there is no application requirement for a specific attribute name. If the definition of an inherited attribute must be specialized in the MIM and there is no application requirement for an attribute name to be given, a subtype may be created and additional textual definitions are created in the MIM for inherited attributes. This entity would then be used to represent the semantics given in the textual definitions of the inherited attributes.

An example of this second practice may be found in ISO 10303-203. The entity supplied_part_relationship is declared as a subtype of the generic product_definition_relationship entity. Since there are no application requirements for specific attribute names, the definitions are refined to say that the related_product_definition is to be interpreted as that product_definition that is supplied by an outside organization and the relating_product_definition is the product_definition that is internal to the owning organization.

5.3.8 Documenting new MIM constructs

The AM document must include complete definitions of all new MIM constructs found in the MIM EXPRESS short listing. It may also include textual definitions and descriptions from the interfaced resource constructs that have been further specialized to suit the established application context of the AM. Guidance for developing and formatting these definitions is found in *Supplementary directives for the drafting and presentation of ISO 10303*.

5.4 Mapping specification

The mapping specification is a pivotal component of an AM. The mapping specification specifies the relationship between the information requirements as specified in the ARM and the resource constructs that satisfy those requirements in the module interpreted model. Unless otherwise specified in this standing document, the mapping specification for an AM shall follow the *Guidelines for the development of mapping specifications, 2nd edition*.

Exceptions to the mapping specification guidelines include the following.

- In all cases where the guideline uses or specifies the term application interpreted model or AIM, that term is to be replaced by module interpreted model or MIM.

- As AM ARMs are written using EXPRESS, the concept of application assertion is not applicable. These are specified in the EXPRESS structure rather than separately from the application object.

- In all cases where options are allowed to be specified in the mapping specification that may be numbered, they shall be numbered. This allows options to be referenced by number in using application modules.

- An application module may be the source of an element in the mapping specification.

Several concepts are available to enable generic and abstract concepts to be defined in one AM but completed in a using AM.

- Incomplete or uninstantiable ARM concepts shall appear in the mapping specification but need not be fully mapped in the mapping specification. In the case that the concept is an abstract supertype in the ARM with the intent that using AMs will define subtypes, text may be specified stating that the mapping appears in using AMs.

- All concepts that appear in the mapping specification need not appear in the MIM. This enables the constraint on a mapping to be specified in one AM while allowing the MIM object into which something is mapped to appear in the MIM of a using AM.

- The mapping specification in a using AM may eliminate one or more of the options found in a supertype declared in a required AM. This is specified in text referencing the mapping in the used AM and stating the number of the mapping that is eliminated. An example may be added that is the referenced mapping specification repeated in the AM with the eliminated option omitted.

5.5 Short names table

Short names are required for each entity in the MIM EXPRESS short form schema. An explanation of short names is found in *Guidelines for the development and approval of STEP application protocols*.

Annex A

Conformance testing concepts for application modules

The application concepts defined in application modules are of varying levels of detail, scope, semantic completeness and applicability to specific contexts. For these reasons not all application modules have the same requirements on conformance testing. The following guidelines and principles shall be used to determine the required testing related developments for application modules.

- all application modules shall have associated ISO 10303-21 files covering every application object defined in the application module which maps to a construct in the MIM;
- application modules that are the data specification for an application protocol shall fall under the same requirements for providing conformance testing supporting documents as a non-modularized AP.

Annex B

Examples

This annex contains examples of a UoF, application objects, and relationship attributes that appear in the example mapping specification.

B.1 Example UoF, application objects, and relationship attributes that appear in the example mapping specification

4.1 Units of Functionality

Product_identification specifies things organizations have identified for some purpose. This UoF consists of the following application objects:

- product;
- product_category;
- organization.

4.2 ARM entity definitions

(*

4.2.1 product application object

A **product** is something an **organization** has identified for some purpose.

Products are uniquely identified by product.id, organization.id and one or more product_category.name.

The mechanism for guaranteeing the uniqueness of organization id is outside the scope of ISO 10303.

The same organization that assigns the product id also categorizes the product in the context of uniquely identifying the product.

EXPRESS Specification

```
*)
ENTITY product;
id : label;
name : text;
categories : SET[1:?] product_category;
id_assigning_organization : organization;
WHERE
```

```
wr1: id_assigning_organization :=: categories[1].name_assigning_organization;
END_ENTITY;
(*
```

---

id : the identification of the product assigned by the organization
name : words by which the product is known.
categories : the categories by which the product is classified or which specify the type of the product
id_assigning_organization : the organization that assigned the product its id
*)

(*

4.2.2 product_category application object

A product_category is a possible classification for products by an organization.

A product_category is uniquely identified by its name within the organization.

The same organization that assigns the product.id also categorizes the product in the context of uniquely identifying a product.

EXPRESS Specification

```
*)
ENTITY product_category;
name : label;
name_assigning_organization : organization;
END_ENTITY;
(*
```

---

name : the identification of the product_category assigned by the organization
name_assigning_organization : the organization that assigned the product_category its name
*)

(*

4.2.3 organization application object

An **organization** is a group of people involved in a particular business process. An organization is not an ad hoc organization, such as project teams, meetings or informal groups.

The mechanism for guaranteeing the uniqueness of organization id is outside the scope of ISO 10303.

EXPRESS Specification

```
*)
ENTITY organization;
id : label;
name : text;
address : OPTIONAL text;
UNIQUE
organizations_are_unique : id;
END_ENTITY;
(*
```

id : the unique identification of the organization
name : the name of the organization
address : the optional address of the organization which may be used to indicate an item delivery address, a postal address, or a physical location visitor address.
*)

B.2 Example mapping specifications

This annex contains an example mapping specification.

Application element	MIM element	Source	Rules	Reference Path
PRODUCT	product	41		
id	product.id			
name	product.name			
categories #1 relationship to Product_category	PATH			product <- product_related_product_category.products[i] product_related_product_category <= product_category
id_assigning_organization #2 relationship to Organization	PATH			product <- (applied_organization_assignment.items[i] applied_organization_assignment <= organization_assignment {organization_assignment.role ->> organization_role organization_role.name = 'id owner'} organization_assignment.assigned_organization->> organization) (applied_person_and_organization_assignment.items[i] applied_person_and_organization_assignment <= person_and_organization_assignment { person_and_organization_assignment.role ->> person_and_organization_role person_and_organization_role.name = 'id owner'} person_and_organization_assignment.assigned_person_and_organization->> person_and_organization person_and_organization.the_organization->> organization)
PRODUCT_CATEGORY	product_category	41		
name	product_category.name			
id_assigning_organization #1 relationship to Organization	PATH			product_category => product_related_product_category product_related_product_category.products[i] ->> product <-

				(applied_organization_assignment.items[i] applied_organization_assignment <= organization_assignment {organization_assignment.role ->> organization_role organization_role.name = 'id owner'} organization_assignment.assigned_organization->> organization) (applied_person_and_organization_assignment.items[i] applied_person_and_organization_assignment <= person_and_organization_assignment person_and_organization_assignment.assigned_person_and_organization->> person_and_organization person_and_organization.the_organization->> organization)
ORGANIZATION	organization	41		
id	organization.id			
name	organization.name			
address	organizational_address	41		organization <- organizational_address.organizations[i] organizational_address

B.3 Example scope refinement

There is a requirement for an AP team to be able to use a portion of an existing AM whose scope is too large. The concept of “scope refinement” is intended to address this requirement. The process is defined as:

- Break the existing AM into two or more smaller AMs;
- Exactly the same scope for the combination of the smaller AMs as for the existing AM;
- No change to the Application objects or mappings allowed;
- Need to allow APs using the existing AM to be unaffected.

The following application objects define the concept of a **product** and a **product_version** in a single **product_id** AM. The **pdm_ap** then uses this **product_id** AM in its entirety.

```
*)

SCHEMA product_id_arm;

ENTITY product;
id : label;
END_ENTITY;

ENTITY product_version;
of_product : product;
END_ENTITY;

END_SCHEMA;
```

```
SCHEMA pdm_ap_arm;

USE FROM product_id_arm;

END_SCHEMA;

(*
```

After the **product_id** AM and the **pdm_ap** AP have been developed, the **process_plant** AP begins development and has the requirement for a product without an associated **product_version**. Thus, the **process_plant** AP team needs to refine the scope of the **product_id** AM creating the **product_only** and **product_version** AMs.

```
*)

SCHEMA product_only_arm;

ENTITY product;
id : label;
END_ENTITY;
```

```
END_SCHEMA;
```

```
SCHEMA product_version_arm;

USE FROM product_only_arm;

ENTITY product_version;
of_product : product;
END_ENTITY;

END_SCHEMA;
```

```
SCHEMA plant_ap_arm;

USE FROM product_only_arm;

END_SCHEMA;
```

(*

Finally, the **product_id** AM needs to be revised to use these two new AMs. The **pdm_ap** need not be revised as the scope refinement maintains the original scope of the **product_id** AM.

*)

```
SCHEMA product_id_arm;

USE FROM product_only_arm;

USE FROM product_version_arm;

END_SCHEMA;
```

B.4 Example management resource completion

The following example illustrates the use of the concept completion and assignment technique in an AIM. The items attribute of the new SUBTYPE **applied_date_assignment** that references the new SELECT type **date_assigned_items** illustrates the only case where it is allowable to add an attribute in a subtype.

*)

SCHEMA resource_example_schema;

```
REFERENCE FROM date_schema (date, date_role);

ENTITY date_assignment;
ABSTRACT SUPERTYPE;
role : date_role;
assigned_date : date;
END_ENTITY
```

```
END_SCHEMA; -- resource_example_schema

SCHEMA concept_completion_example_schema;

USE FROM partial_product_definition_schema
(product,
product_definition_formation);

USE FROM resource_example_schema (date_assignment);

TYPE date_assigned_items = SELECT -- SELECT type definition
(product,
product_definition_formation);
END_TYPE;

ENTITY applied_date_assignment; -- ENTITY subtype definition
SUBTYPE OF (date_assignment);
items : SET [1:?] OF date_assigned_items;
END_ENTITY;

END_SCHEMA; -- concept_completion_example_schema

(*
```

The new subtype entity declaration may also be combined with the localization of constraints practice which would enable the specification of either behavioral or referential integrity constraints in the WHERE clause of the entity. An example of this is provided in the following example schema; date assignments are coordinated with the value of the role attribute. The function date_time_correlation says, for example, that a date with the role of "creation date" must be assigned to a product_definition entity. Again, the addition of attributes in an newly defined subtype is allowed only for the assignment template structures in the STEP integrated resources; these ABSTRACT SUPERTYPE entities are incomplete by definition and must be completed in the using schema.

```
*)

SCHEMA role_correlation_example;

TYPE date_and_time_assigned_items = SELECT
(product_definition,
change_request,
start_request,
change,
start_work,
approval_person_organization,
contract,
security_classification,
certification);
END_TYPE; -- date_time_item

ENTITY applied_date_and_time_assignment
SUBTYPE OF (date_and_time_assignment);
items : SET [1:?] OF date_and_time_assigned_items;
WHERE
WR1: date_time_correlation(SELF);
END_ENTITY; -- applied_date_and_time_assignment
```

```
ENTITY date_and_time;
date_component : date;
time_component : local_time;
END_ENTITY; -- date_and_time

ENTITY date_and_time_assignment
ABSTRACT SUPERTYPE;
assigned_date_and_time : date_and_time;
role : date_time_role;
END_ENTITY; -- date_and_time_assignment

ENTITY date_time_role;
name : label;
END_ENTITY; -- date_time_role

FUNCTION date_time_correlation
(e : applied_date_and_time_assignment ) : BOOLEAN;

LOCAL
dt_role : STRING;
END_LOCAL;

dt_role := e\applied_date_and_time_assignment.role.name;

CASE dt_role OF

'creation_date' : IF SIZEOF (e.items) <>
  SIZEOF (QUERY (x <* e.items |'ROLE_CORRELATION_EXAMPLE.'   +
'PRODUCT_DEFINITION' IN TYPEOF (x)))

  THEN RETURN(FALSE);

  END_IF;

'request_date' : IF SIZEOF (e.items) <>
SIZEOF (QUERY (x <* e.items |
SIZEOF (
['ROLE_CORRELATION_EXAMPLE.CHANGE_REQUEST' +
'ROLE_CORRELATION_EXAMPLE.START_REQUEST'] *
TYPEOF (x)) = 1))

THEN RETURN(FALSE);

END_IF;

'release_date' : IF SIZEOF (e.items) <>
SIZEOF (QUERY (x <* e.items |
SIZEOF (
['ROLE_CORRELATION_EXAMPLE.CHANGE' +
'ROLE_CORRELATION_EXAMPLE.START_WORK'] *
TYPEOF (x)) = 1))

THEN RETURN(FALSE);

END_IF;

'start_date' : IF SIZEOF (e.items) <>
SIZEOF (QUERY (x <* e.items |
```

```
SIZEOF (
['CONFIG_CONTROL_DESIGN.CHANGE' +
'ROLE_CORRELATION_EXAMPLE.START_WORK'] *
TYPEOF (x)) = 1))

THEN RETURN(FALSE);

END_IF;

'sign_off_date' : IF SIZEOF (e.items) <>
SIZEOF (QUERY (x <* e.items |
'ROLE_CORRELATION_EXAMPLE.' +
'APPROVAL_PERSON_ORGANIZATION'
IN TYPEOF (x)))

THEN RETURN(FALSE);

END_IF;

'contract_date' : IF SIZEOF (e.items) <>
SIZEOF (QUERY (x <* e.items |
'ROLE_CORRELATION_EXAMPLE.CONTRACT'
IN TYPEOF (x)))

THEN RETURN(FALSE);

END_IF;

'certification_date' : IF SIZEOF (e.items) <>
SIZEOF (QUERY (x <* e.items |
'ROLE_CORRELATION_EXAMPLE.CERTIFICATION'
IN TYPEOF (x)))

THEN RETURN(FALSE);

END_IF;

'classification_date' : IF SIZEOF (e.items) <>
SIZEOF (QUERY (x <* e.items |
'ROLE_CORRELATION_EXAMPLE.' +
'SECURITY_CLASSIFICATION'

IN TYPEOF (x)))

THEN RETURN(FALSE);

END_IF;

'declassification_date' : IF SIZEOF (e.items) <>
SIZEOF (QUERY (x <* e.items |
'ROLE_CORRELATION_EXAMPLE.' +
'SECURITY_CLASSIFICATION'
IN TYPEOF (x)))

THEN RETURN(FALSE);
END_IF;

OTHERWISE : RETURN(TRUE);
END_CASE;
```

```
RETURN (TRUE);
END_FUNCTION; -- date_time_correlation

END_SCHEMA; -- role_correlation_example

(*
```

B.5 Entity behavioral constraints

An example of the use of entity behavioral constraints consists of an application requirement defined in an ARM for two different types of the product_definition entity. One type of product_definition entity is always a component in an assembly and another type of product_definition is never a component in an assembly. This is a behavioral constraint on the product_definition that could be implemented with two subtypes or with constrained attribute values. In the case where subtypes are created, the first subtype could contain a constraint that says it must always be used in the product_definition_relationship entity as the related_product_definition. The second subtype of product_definition could contain a constraint specifying that it shall never be used in the product_definition_relationship entity. In the case where attribute values are constrained, a global rule could be written specifying that when the description attribute of product_definition has a value of component, the product_definition must always be used in a product_definition relationship as the related_product_definition.

In the case that subtypes are created, entity behavioral constraints are specified as local rules and the EXPRESS USEDIN function is employed to gain access to the other entities that reference the particular entity which needs to be constrained in the MIM. Each subtype definition allows the MIM schema to specify different uses of the generic concept (as defined in the STEP integrated resources) for different purposes (as defined in an ARM).

B.6 Referential integrity constraints

For example, referential integrity constraints would be used to support an application requirement for two separate uses of the polyline entity defined in ISO 10303-42. Let us assume for the case of this example, that the requirements for the usage of the polyline are differentiated by the fact that there are two different mathematical methods for describing the shape of something. One representation requires polylines to contain exactly two points for the representation of line segments. The other representation has a requirement for polylines to contain more than two points and line segments defined by trimmed curves with underlying lines as the basis curves. This example will use the entities defined in ISO 10303-41, ISO 10303-42 and ISO 10303-43.

Since the polyline is constrained differently based on its usage in the particular method, if both methods are required in a single AP, the constraints on use of the polyline entity must be localized. Constraint localization is accomplished by defining an entity in the MIM schema that is a subtype of a resource entity to define a scope for the constraints. To localize the constraints in this example, two subtypes of the shape_representation entity from ISO 10303-41 need to be created where the applicable constraints for the polyline are specified in the two different representation subtypes. One of them is to create a scope for the mathematical method in which polylines are defined only by two points; and the other is to create a scope for the mathematical method in which polylines are defined by more than two points. The entity definition for the

subtypes shall contain an explanation of the purpose of the constraints. The polyline entity is referenced by an attribute of the geometric_set entity, which is, in turn, referenced by the representation structure from ISO 10303-43. The constraints on polyline will be specified as referential integrity constraints on the representation_items (inherited by shape_representation entity from the representation entity in ISO 10303-43) that are of type "polyline" within the contents set of the geometric_set that is in the set of items in the representation entity. In one shape_representation subtype, the size of the set of points that define the polyline is constrained to two elements, and in the other shape_representation subtype the size of the set of points that define the polyline is constrained to be more than two. Each MIM entity (the created subtypes of the shape_representation entity), therefore, defines a context within which conflicting constraints on the polyline may exist within the MIM schema.

Referential integrity constraints are written as local rules that use the EXPRESS TYPEOF function to identify the appropriate traversal through the reference path in specifying the constraint on the subtype and gain access to the attributes that ultimately need to be constrained. In the example, each subtype allows different constraints to be placed on a single attribute of a single entity depending on the reference path by which those attributes are reached.